



AFRL-RY-WP-TR-2014-0016

REAL NUMERICAL ALGEBRAIC GEOMETRY: FINDING ALL REAL SOLUTIONS OF A POLYNOMIAL SYSTEM

**Daniel J. Bates, Daniel A. Brake, Wenrui Hao, Jonathan D. Hauenstein, Andrew J. Sommese,
and Charles W. Wampler**

Colorado State University

**FEBRUARY 2014
Final Report**

Approved for public release; distribution unlimited.

See additional restrictions described on inside pages

STINFO COPY

**AIR FORCE RESEARCH LABORATORY
SENSORS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

AFRL-RY-WP-TR-2014-0016 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

*//Signature//

JARED CULBERTSON, Program Manager
Electro-Optic Exploitation

//Signature//

CLARE MIKULA, Branch Chief
Electro-Optic Exploitation

//Signature//

DOUG HAGER, Deputy
Layered Sensing Exploitation Division

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

*Disseminated copies will show “//Signature//” stamped or typed above the signature blocks.

REPORT DOCUMENTATION PAGE					Form Approved OMB No. 0704-0188	
<p>The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</p>						
1. REPORT DATE (DD-MM-YY) February 2014		2. REPORT TYPE Final		3. DATES COVERED (From - To) 25 October 2012 – 25 October 2013		
4. TITLE AND SUBTITLE REAL NUMERICAL ALGEBRAIC GEOMETRY: FINDING ALL REAL SOLUTIONS OF A POLYNOMIAL SYSTEM				5a. CONTRACT NUMBER FA8650-13-1-7317		
				5b. GRANT NUMBER		
				5c. PROGRAM ELEMENT NUMBER 61101E		
6. AUTHOR(S) Daniel J. Bates, Daniel A. Brake, Wenrui Hao, Jonathan D. Hauenstein, Andrew J. Sommese, and Charles W. Wampler				5d. PROJECT NUMBER 1000		
				5e. TASK NUMBER 11		
				5f. WORK UNIT NUMBER Y0VW		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Colorado State University 601 S. Howes Street Fort Collins, CO 80521-2807				8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory Sensors Directorate Wright-Patterson Air Force Base, OH 45433-7320 Air Force Materiel Command United States Air Force				10. SPONSORING/MONITORING AGENCY ACRONYM(S) AFRL/Ryat		
				11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S) AFRL-RY-WP-TR-2014-0016		
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.						
13. SUPPLEMENTARY NOTES <p>This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This material is based on research sponsored by Air Force Research Laboratory (AFRL) and the Defense Advanced Research Agency (DARPA) under agreement number FA8650-13-1-7317. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory (AFRL) and the Defense Advanced Research Agency (DARPA) or the U.S. Government.</p>						
14. ABSTRACT <p>This project resulted in software and algorithms for computing the real solution of polynomial systems contained within complex curves and surfaces. The primary product was Bertini Real, a publicly available software package for these computations. Secondary products included four articles, all in progress. These articles document the algorithms developed during the reporting period, including (1) a complete algorithm in the case of complex surfaces with no restrictions on the type of surface; (2) a technical article on the theory underlying and algorithm for finding critical points; (3) an article on Bertini Real; and (4) an article on the real numerical irreducible decomposition.</p>						
15. SUBJECT TERMS <p>real cellular decomposition, numerical algebraic geometry, polynomial system, real algebraic geometry, homotopy continuation, Bertini</p>						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT: SAR	18. NUMBER OF PAGES 22	19a. NAME OF RESPONSIBLE PERSON (Monitor) Jared Culbertson 19b. TELEPHONE NUMBER (Include Area Code) N/A	
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified				

Table of Contents

Section	Page
List of Figures	ii
List of Tables	ii
1.0 Summary	1
2.0 Introduction.....	2
2.1 Background on the Problem.....	2
2.2 Cellular Decomposition	2
2.3 Computing a Cellular Decomposition	3
3.0 Methods, Assumptions, and Procedures	5
4.0 Results and Discussions.....	6
4.1 Software	6
4.1.1 Description.....	6
4.1.2 Curve Decompositions with Bertini Real	6
4.1.3 Surface Decompositions with Bertini Real.....	7
4.1.4 Challenges.....	10
4.1.5 Other Tools Produced	10
4.1.6 Obtaining Bertini Real	10
4.2 Mathematical Results Supporting Software.....	10
4.2.1 Algorithms for 1-D, 2-D, 3-D Real Cellular Decomposition	10
4.2.2 Finding the Critical Points of the Critical Curve	11
4.2.3 Real Numerical Irreducible Decomposition	11
4.3 Performance on Proposed Tasks.....	12
4.3.1 Overview.....	13
4.3.2 More Detailed Comments	13
5.0 Conclusions.....	14
6.0 References.....	15
List of Acronyms, Abbreviations, and Symbols	16

List of Figures

Figure	Page
1. Intersection of the Eistute System with a Sphere	7
2. Cellular Decomposition of a Sphere	8
3. Cellular Decomposition of a Torus.....	8
4. The Barth Sextic System, as Decomposed by Bertini Real	9
5. A Theorem Yielding a Computationally Feasible Approach.....	11

List of Tables

Table	Page
1. Final Status of Project Tasks	13

1.0 Summary

This project is centered around the problem of computing all real points, curves, and surfaces embedded in 1 dimensional (1-D) and 2 dimensional (2-D) complex solution sets of polynomial systems. Polynomial systems are sets of equations that can be used to model a variety of applications, ranging from robotic arms to biochemical reaction networks to vehicle stability control problems.

There are currently excellent methods that find all complex solution sets (many continuation-based methods are implemented in Bertini, created by four of the six team members), but there was previously no known way to extract the real solutions sets from the complex ones, except in a few very basic cases. The difficulty is that real solutions (points, curves, surfaces, etc.) can be hidden within higher-dimensional complex solution sets, and there seemed to be no straightforward way to dig them out.

This project focused on the development and implementation of methods to do exactly this: compute all real solutions for polynomial systems, embedded in complex components of dimension one or two.

The primary deliverable from this project is the publicly available software package Bertini Real (BR). Various mathematical results were necessary to make this software package possible. All outcomes are described in Section 4.0. Performance on particular tasks from the original proposal are summarized in the last part of Section 4.0, with the few specific shortcomings detailed at the very end of that section.

2.0 Introduction

2.1 Background on the Problem

It is a fundamental problem to be able to compute and manipulate the real solution set of polynomial systems. Much of classical mathematics deals with one form or another of this problem. Problems of pressing importance including mechanical engineering, graphics, numerical solution of systems of differential equations, and optimization would be significantly impacted by algorithms and software effectively solving this problem.

Possibly the oldest approach to understanding geometric objects is by slicing: see the discussion of the Apollonius method in [7]. At the end of the 19th century this led to early versions (by Castelnuovo and Enriques) of what are now called the First and Second Lefschetz Theorems [10]. These theorems give a prescription to rebuild the homology of a projective manifold in terms of the homology of a general section and the homology of the singular sections of the manifold belonging to a general one-parameter family of hyperplane sections.

In modern times, a Morse theory approach to this problem [1,2,6] initiated from a lecture by R. Thom led to much more general versions of the Lefschetz Theorems, and ultimately a Morse theory for singular algebraic sets [8].

In the computational algebra community the same approach led to cylindrical decompositions and structures such as roadmaps [3]. The literature on this is vast, but the algorithms have so far been unable to deal with all but the simplest cases.

This projects back to the original inspiration for all of these developments: a pencil (i.e., a linear parameter family) of hyperplane sections. Using the powerful, efficient, and fast algorithms of numerical algebraic geometry, the approach by pencils of hyperplane sections may be used to give a numerically natural decomposition into cells.

2.2 Cellular Decomposition

The general idea of a cell decomposition is easiest to grasp by first considering piecewise linear sets such as polygonal curves, polyhedral surfaces, solid polyhedra, and, more generally, polytopes in higher dimensions. For a polygonal curve, one needs only to list the vertices (i.e., store a numerical value for each coordinate of each vertex) and then list the edges, each one consisting of a pair of integers indicating which vertices are the endpoints of the line segment. Since the edge is linear, this representation suffices to represent the whole edge. For a polyhedral surface with convex polygonal faces, this generalizes to a list of vertices, a list of edges, and a list of faces. General polyhedral surfaces can be so represented by first subdividing any nonconvex face into a union of convex pieces. It is typical to do so by breaking each face, even convex ones, into a set of triangles. This is common in computer graphics, where specialized hardware and software is designed to rapidly process huge numbers of triangles in parallel fashion on graphics cards. For polyhedral solids, one lists a union of 3-dimensional cells, each a convex polyhedron, in terms of the faces that form the boundary of the cell, each of which has bounding edges, each of which terminates in vertices. All data is integer except for the numerical approximations of all the vertices. Polytopes in higher dimensions follow a similar scheme.

When one wishes to represent a nonlinear curve or surface, etc., such as might be contained in

the real solutions of a polynomial system, one common approach is to approximate the set with a nearby piecewise linear one. This has two deficits:

- (1) it can take a dense set of linear pieces to approximate well the smooth set, a problem that grows exponentially more severe with the dimension of the set, and
- (2) the piecewise linear approximation does not necessarily contain enough information to produce a more accurate approximation to the smooth set, should that be required in subsequent work with the set.

The approach to cell decomposition that we use addresses both these deficits. It represents the smooth set with a sparse set of cells that retains all the information needed to refine the cells to as fine a level as might be desired. Each cell is represented by a similar hierarchy of vertices, edges, faces, etc., as in the piecewise linear case, except the edges, faces, and so on, are all algebraic, and hence potentially nonlinear. It is convenient to call vertices “0-cells,” edges “1-cells,” faces “2-cells,” and so on.

Without explicating all the details, the extra information stored in a k -cell, besides a list of pointers to the $(k-1)$ -cells that form its boundary, is:

- * a polynomial system, say g , which has a k -dimensional irreducible component of which the k -cell is a piece,
- * a real projection, and
- * a numerical approximation of a general point, say w , in the interior of the cell.

Instead of the convexity requirement placed on the cell decomposition of polyhedra, we require that:

- * the k -cell is homeomorphic to a k -dimensional ball (usually called a line segment for $k=1$ and a disk for $k=2$), and
- * there exists a homotopy function $h(x; t) = 0$, so that for any connected 1-real-dimensional path in the unit box, starting at w remains in the k -cell, is nonsingular in the interior of the box, and approaches the boundary of the face whenever approaches the boundary of the unit box.

In the piecewise linear case, a decomposition of a polytope into convex cells, any point in the interior of a cell can be expressed as a convex linear combination of the vertices of the cell. This allows one to easily generate new points in the cell or subdivide the cell into smaller cells. Similarly, the homotopy function associated to a cell decomposition of a real algebraic set provides an equivalent capability to move at will within the cell and to subdivide the cell. In applications, such as to visualize a real algebraic set (or a projection of it to three dimensions) or to build a finite element mesh on it, one may subdivide to whatever resolution necessary to well-approximate the smooth set with a faceted discretization.

2.3 Computing a Cellular Decomposition

Many details on the computation of a cellular decomposition may be found in the new book [4] or the article [5]. Furthermore, some details of the method are described in Section 4A, about the software BR. This is a long, complicated technique, so the details are best left to the references.

However, to illustrate the idea, let us consider the unit circle in the plane, using projection to the x-axis. By solving some polynomial system, we may find the critical points with respect to the projection, namely $(1,0)$ and $(-1,0)$, having projection values -1 and 1 , respectively. Slicing between these two projection values, i.e., at $x=0$, we find two “interior” points of edges, namely $(0,1)$ for the top edge and $(0,-1)$ for the bottom edge. By walking along the edges (via a homotopy), we discover the left and right endpoints (among the critical points) for each edge, thereby completing our initial cellular decomposition. The result is a list of two edges, each with two endpoints and a midpoint.

We can then refine an edge by choosing more gridpoints (not necessarily uniformly) between the two endpoints of the edge and using the homotopy to track along the edge away from the midpoint.

This idea generalizes to surfaces and beyond. As an example of a surface, take the unit sphere in the plane, with a projection down to the xy-plane. The critical set is now the equator, a curve. This curve needs to be decomposed via a second projection, say to the x-axis, yielding the same diamond described in the case of the unit circle above. Looking over the origin, we find two points (generic points for the top and bottom faces, much like the midpoint of an edge) and can represent each face as a 2-cell with a generic point and 1-cells along the boundary.

One major complication is that these geometric objects need not be smooth. One major outcome of this project is that there is now a way to handle *any* curve or surface in any dimension, not just those that are particularly nice.

3.0 Methods, Assumptions, and Procedures

As this was a project in computational mathematics, there is little to say here.

Subsets of the group met several times in various locations to discuss algorithms. Brake (then a postdoc at Colorado State) and Hao (then a graduate student at Notre Dame) then spent much time working on software development while the more senior collaborators worked out details and theory. Bates supervised Brake and Sommesse supervised Hao.

4.0 Results and Discussion

This section includes a description of the software package BR in some detail, followed by brief discussions about algorithms and theory and, finally, a task-by-task analysis of performance.

4.1 Software

4.1.1 Description

BR is the main program produced during this project. Written in C++, it uses the homotopy continuation solver Bertini and several free libraries, most notably Boost, to produce cellular decompositions for real embedded curves and surfaces for components of appropriate dimension for algebraic varieties. BR is a command line tool compilable in OSX and *nix environments, and uses a few other custom utilities to produce a plot or STL file from a polynomial system.

The main input for BR is a Bertini input file. First, the user performs a numerical irreducible decomposition by calling tracktype 1 in Bertini. As the output from this step, a witness_data file, is not human readable, and is difficult to parse, a utility called data2set is provided for the user to split the witness_data file into several witness_set files, which are required for BR. This file contains only the relevant points, linears, and patches, for an individual component -- of which only one will be studied at a time in BR.

The main program is called from the command line as bertini_real. It has many options available at runtime via flags, such as -q for quick mode. At this point, the program runs without further human input, until decomposition is complete.

BR automatically detects the dimension of the component, and enters the appropriate mode for either a curve or a surface. It begins by determining whether the component is self-conjugate, and performing isosingular deflation (if necessary). This ensures that the remainder of the algorithm will function properly.

4.1.2 Curve Decompositions with Bertini Real

The curve algorithm first finds critical points, by performing a left-nullspace singularity calculation. That is, given the original system f , BR finds points such that $\{Jf, \pi\}$, the Jacobian of the system together with the random real projection vector used to compute the decomposition, is singular. As a matter of irrelevant consequence, it also finds left singular vectors.

Between each of the real critical points found, BR slices to find real midpoints, from which it tracks left and right, relative to the particular projection π , to obtain the set of edges for the decomposition. Finally, BR will merge away superfluous edges, if requested (and by default).

A subsampler for refinement of curves is provided, as a separate callable program titled sampler. From the midpoint of each edge, BR tracks to user-supplied tolerance, such that successive points on each edge are no further apart than the tolerance. Hence, we can turn blocky curve decompositions into smooth figures with little effort. See Figure 1 for an illustration of the subsampler. The curve is of degree 12, and consists of many edges. On the left is the raw skeleton of edges produced by BR, and on the right is a refinement of the decomposition, to

a tolerance of 0.1 distance units. Curve decomposition is integral to the surface decomposer, and was the first problem tackled.

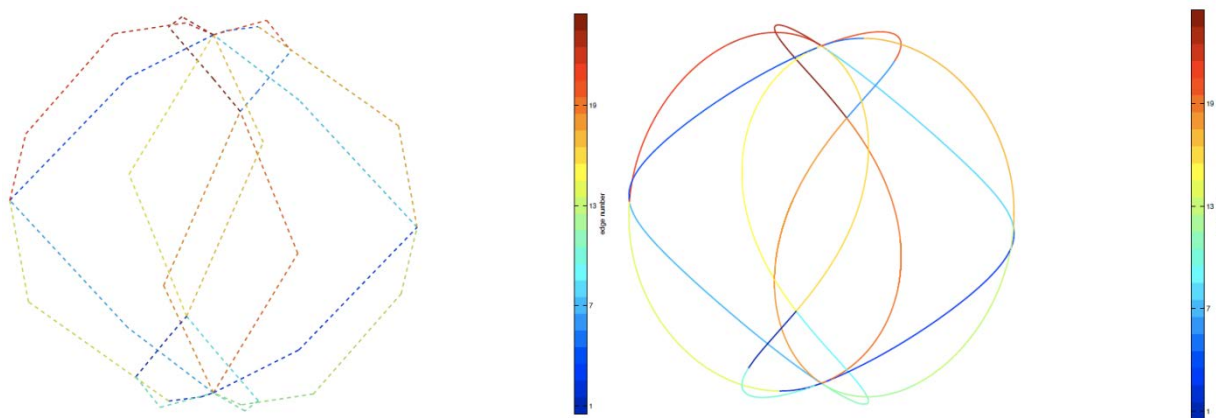


Figure 1. Intersection of the Eistute System with a Sphere

4.1.3 Surface Decompositions with Bertini Real

BR will also decompose real embedded surfaces for components of dimension two. The result is a skeleton of the surface, with a network of connected faces, with midpoints and a joining of edges of curves.

The surface decomposer in BR depends on the curve decomposer, in that the main computed object is the critical curve, with respect to the two random real projections chosen for the computation. After BR computes the critical points for the critical curve, and performs a curve decomposition on it, the program will intersect the surface with an automatically generated sphere of appropriate center and radius, such that all the important information is contained therein. The user can also supply their own sphere of interest.

Having the critical and sphere intersection curves, BR enters a slicing routine, wherein between and at each critical point for each of the critical and sphere curve, a merged curve decomposition is produced.

The final step is to connect the dots, so to speak. The midpoint for each face is the midpoint for an edge of a mid-slice, and we need to determine to which critical slices the midpoint connects. Using a custom written solver, using Bertini as the underlying tracker, BR tracks simultaneously three points on three different systems. Having mapped face connections, the surface decomposition is complete.

A sampler refinement method is under development at this time, with the ability to refine the triangulation formed by the initial decomposition to arbitrary tolerances chosen by the user at runtime.

Perhaps the simplest example of a surface decomposition is provided by the unit sphere. The decomposition consists of two faces, top and bottom, joined at the waistline of the sphere by the critical curve along the equator relative to the projections used. See Figure 2. The result has two faces, in blue and red, along with the critical curve along a great circle on the surface.

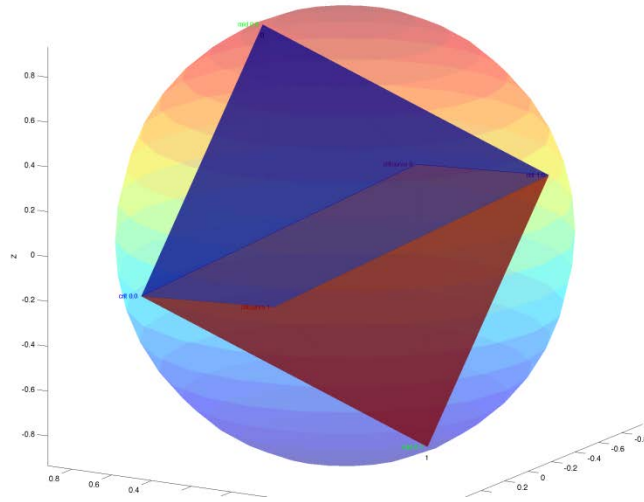


Figure 2. Cellular Decomposition of a Sphere

The torus is a simple non-trivial surface example. Having a single hole in the center, the critical slices at the ends of the hole produce non-degenerate curve decompositions, so there is actual work for BR when connecting the midpoints of faces. The critical curve consists of two disjoint pieces, and typically eight edges. There are four critical slices, and three midslices, yielding a total of 8 faces. See Figure 3, with $r=0.5$, $R=2.0$, so that $(x^2 + y^2 + z^2 + R^2 - r^2)^2 - 4R^2(x^2 + y^2) = 0$.

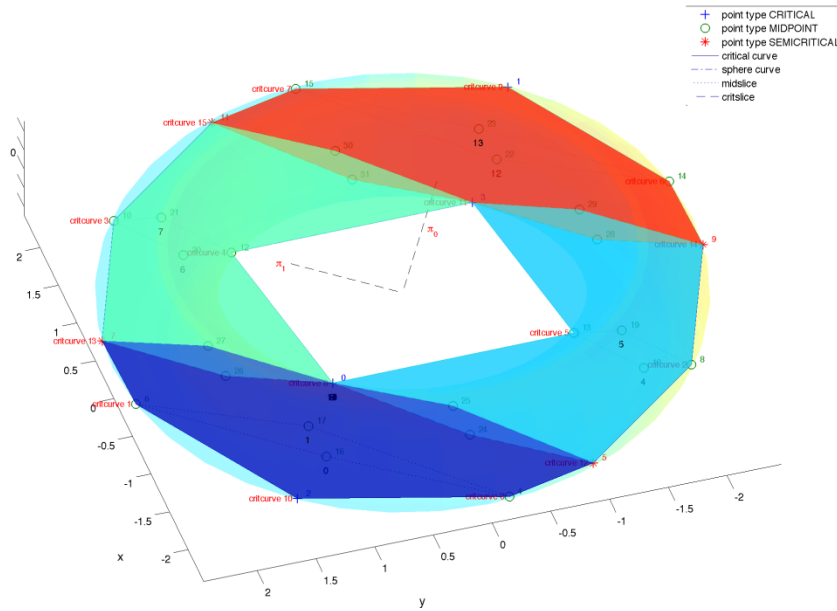


Figure 3. Cellular Decomposition of a Torus

Decomposed with respect to a random set of two real projections, $\pi_{1,2}$ as shown, it has 8 faces, and the critical curve has two disjoint pieces. Produced in under 20 seconds on a single processor.

A much more involved example is the Barth Sextic system, which can be sourced from <http://cage.ugent.be/~hs/barth/barth.html>. The degree 6 surface is defined over the complex numbers, and obtaining a full 3-D model of the system has been challenging until now. Previously, animations or renderings were created using ray-tracing software such as the famous and robust POV-Ray. Now, however, we have a 3-D model of the system, and for the first time can produce an STL model suitable for 3-D printing, in a reasonable amount of time. The initial decomposition in Figure 4 took approximately 3.5 hours on four cores on a laptop computer. A cluster can perform the decomposition much quicker, and the program has been further parallelized since this particular decomposition was obtained. Top: On the left is the skeleton of curves, also decomposed by BR, which are connected to each other to make the faces of the right hand side. The system is defined by three spatial variables, and has a total of 65 singular points, with the vertices of two parallel dodecahedrons embedded therein. Bottom: A rendering of an STL file, from which one could 3-D-print the surface.

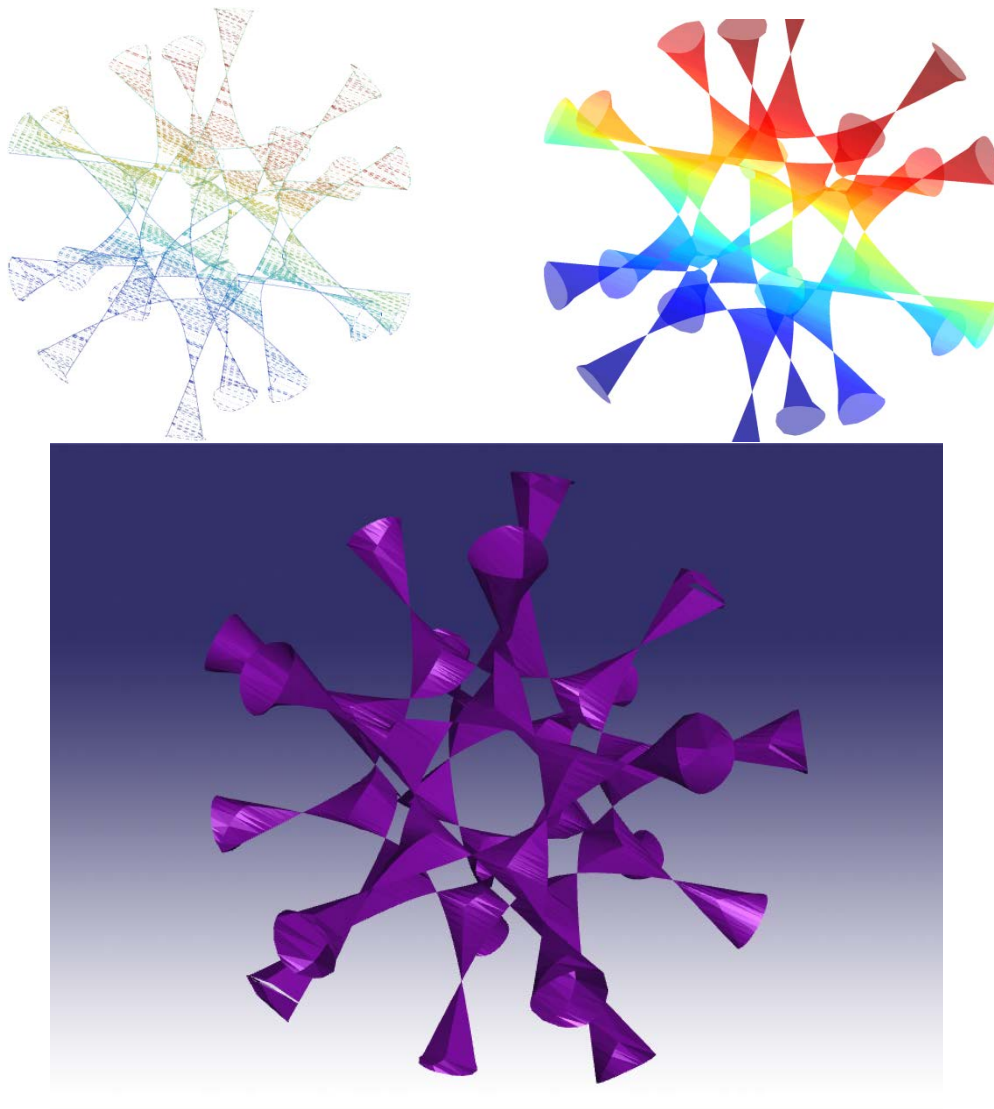


Figure 4. The Barth Sextic System, as Decomposed by Bertini Real

4.1.4 Challenges

BR is a numerical tool, and subject to the natural challenges and issues from this field. Such problems as numerical thresholding, convergence tolerances, etc, arise in BR.

The main hurdles for successful BR decomposition are ensuring that paths track successfully. This requires that the pre-endgame tolerance is set tightly enough, but not too tightly - runtimes become exceedingly long when tolerances are too tight.

4.1.5 Other Tools Produced

The command line tool BR is accompanied by Matlab code to visualize the results. This in turn spurred the development of an alternate human interface through the Leap Motion controller, so that the user can interact with the decompositions on screen without contact with any object, simply by waving fingers around.

The production of STL files for 3-D printing has also been automated. Users may input a polynomial system having a surface as solution set and receive a 3-D model of the surface as output. This seems to be the first software package of this sort.

4.1.6 Obtaining Bertini Real

BR is version controlled via Subversion, and the latest version of the repository is available for checkout to the public without credentials, at svn://paramotopy.com/bertini_real.

Documentation, produced automatically by Doxygen, is available at bertinireal.paramotopy.com.

4.2 Mathematical Results Supporting Software

This project has resulted in (a) 3 algorithms for finding real solutions, (b) a technical result supporting a particular point in these algorithms, and (c) an algorithm for computing the real numerical irreducible decomposition. As all three of these products will appear in publications, we provide only overviews of them here.

4.2.1 Algorithms for 1-D, 2-D, 3-D Real Cellular Decomposition

Previously, there were known algorithms for decomposing the real solutions within a complex curve (in general) and those within a complex surface (making specific assumptions on the type of surface), but no implementation of any such method.

One outcome of this work is that there are now complete algorithms for decomposing the real solutions within any complex curve or surface, along with the implementation described above. This extension of previously-known methods hinged on the recent development of *isosingular deflation* for desingularizing algebraic sets with generic multiplicity greater than 1. Another (minor) outcome of this work is that isosingular deflation has now been implemented in Matlab and is part of the BR package described above.

Another outcome is work towards an algorithm for real solutions within 3-dimensional complex solution sets. This algorithm is now fully understood and there is progress on writing it down for a publication. There is a fundamental increase in complexity when moving from dimension k to dimension $k+1$ with the methods of this project, and even the 3-dimensional version of the technique is far more technical than expected. Ultimately, while such an algorithm for decomposing the real solutions of dimension N (any N) is feasible, it seems that it will actually be exceedingly difficult to implement correctly.

4.2.2 Finding the Critical Points of the Critical Curve

By far the most complicated issue with the methods of this project is the detection of the critical points on the critical curve of a surface. In particular, given a polynomial system with $N-2$ equations in N variables and two projections, it is necessary to know how to find the critical points (under one projection) of the critical curve given by the other projection. There are a number of approaches to this. One outcome of this project is a proof of the theorem in Figure 5.

Theorem 0.2 *Suppose that $f : \mathbb{C}^n \rightarrow \mathbb{C}^{n-2}$ is a polynomial system such that $S \subset \mathcal{V}(f)$ is an irreducible surface. Let $\pi_1, \pi_2 : \mathbb{C}^n \rightarrow \mathbb{C}$ be two general projections with $\pi = (\pi_1, \pi_2)$. Let C be the curve of critical points of S with respect to π . Then, the critical points of C with respect to π_1 are the points $x \in S$ such that there exists $v \in \mathbb{P}^{n-2}$ with*

$$\begin{bmatrix} f(x) \\ v \cdot \begin{bmatrix} Jf(x) \\ J\pi_1 \end{bmatrix} \end{bmatrix} = 0.$$

Figure 5. A Theorem Yielding a Computationally Feasible Approach

4.2.3 Real Numerical Irreducible Decomposition

A side project from the originally proposed research is the computation of the *real numerical irreducible decomposition* of a polynomial system. This is a numerical irreducible decomposition of the Zariski closure of the real solutions of the system, i.e., a decomposition of those complex solution sets that necessarily contain a real component. This is useful as a preprocessing step to decomposing the real solutions within the (possibly many) complex solution sets of a polynomial system. Indeed, by detecting those complex components with *no* chance of harboring real point, they can be eliminated from subsequent consideration.

The subroutines needed to compute the real numerical irreducible decomposition are nearly identical to those needed for the real cellular decomposition described above. As a result, this computation was a natural corollary to the main part of the project.

Such a decomposition is useful, for example, as an assemblability test in mechanical engineering. Here, the real solutions of the system describe how to assemble a mechanism. Thus, from a real numerical irreducible decomposition, one can decide if there exists an assembly as well as the degrees of freedom of the mechanism once it has been assembled. Such information may not be available from the classical numerical irreducible decomposition since the real and complex dimensions can be different.

One mechanism where this occurs is a cubic-centered 12-bar mechanism, first presented in [11]. This mechanism is described by 17 polynomials in 18 variables so that the complex dimension is at least one. However, there are configurations of the mechanism that are rigid over the real numbers, i.e., isolated real solutions. The computation of the real numerical irreducible decomposition of the polynomial system describing this mechanism identifies those complex components containing real solutions and also picks out some real solutions buried within some of the complex components.

For this example, there are 17 polynomials in 18 variables. Since the complex curves are the nondegenerate components of interest, the 8 irreducible curves are analyzed. Six of them have degree 4 while two have degree 6. Since a computation using the method of Hauenstein in [9] yields a smooth real point on each of the first 6, they are necessarily real radical. That computation also shows that the degree 6 curves have real points, but those points were singular. After further computation, it becomes clear that there are no real smooth points on the two degree 6 curves, so the real numerical irreducible decomposition of the mechanism curves consist of six curves of degree 4 and two points that lie in the intersection of the two degree 6 curves.

4.3 Performance on Proposed Tasks

The originally proposed project consisted of 10 tasks plus one optional task given time. Our performance on each task is indicated in the next section, with specific explanations for incomplete tasks described in the following section. Put simply, the principle goal of this project (methods and software for finding real solutions) was completed, and the only missing pieces are optimizations and automations that turned out to be unnecessary.

4.3.1 Overview

Table 1. Final Status of Project Tasks

Task	Status	Notes
1. Isosingular deflation code	complete	in BR
2. 1-D cellular algorithm	complete	
3. 1-D cellular code	complete	in BR
4. 2-D cellular algorithm	complete	
5. 2-D cellular code	complete	in BR
6. Real NID algorithm	complete	will appear in publication
7. 3-D cellular algorithm	80%	known, not yet written up
8. Increased efficiency, isosingular deflation	adequate	see below
9. Analyzing isosingular deflation approaches	adequate	see below
10. Real NID code	75%	see below
11. 3-D cellular code (optional)	50%	optional; some supporting subroutines in BR

4.3.2 More Detailed Comments

Task 7: It is now clear how this algorithm goes, but it is highly technical and has therefore not yet been written down. The plan is to eventually put this in a publication.

Tasks 8 and 9: While more analysis could be conducted about the efficiency of isosingular deflation, the fact is that the current implementation choices are more than adequate for BR. Isosingular deflation is not a major component of the computational cost of the main algorithms, so there is no practical point in spending time optimizing it.

Task 10: Given the subroutines implemented in BR, it is feasible to compute the real NID in an *ad hoc* manner. However the main algorithm for calling these subroutines has not yet been completed.

Task 11: This task was optional. The intention is to include this code in Bertini 2.0 (along with the rest of BR), once we have built the foundation of that redevelopment of Bertini.

5.0 Conclusions

All told, we developed the methods and software intended, missing only some of the lowest level optimization (which turned out to be unnecessary) and highest level of coding (task 10).

Researchers now have the ability to use software to find real solution sets of polynomial systems embedded in complex curves and surfaces. This was not previously possible in this level of generality. The hope is that this significantly improves the ability of modelers to use polynomial systems in their models, rather than relying so heavily on (not necessarily optimal) linear models.

6.0 References

- [1] A. Andreotti and T. Frankel. The Lefschetz theorem on hyperplane sections. *Ann. of Math.* (2) 69, 713-717, 1959.
- [2] A. Andreotti and T. Frankel. The second Lefschetz theorem on hyperplane sections. 1969 in *Global Analysis (Papers in Honor of K. Kodaira)*, pp. 1-20 Univ. Tokyo Press, Tokyo.
- [3] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*, vol. 10 of Algorithms and Computation in Mathematics, Springer-Verlag, Berlin, second ed. 2006.
- [4] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler. *Numerically solving polynomial systems with Bertini*, SIAM, 2013.
- [5] G.M. Besana, S. Di Rocco, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler, Cell decomposition of almost smooth real algebraic surfaces, preprint, 2012.
- [6] R. Bott. On a theorem of Lefschetz. *Michigan Math. J.* 6, pp. 211-216, 1959.
- [7] T. Fujita. *Classification theories of polarized varieties*. Volume 155 of London Mathematical Society Lecture Note Series. Cambridge University Press, Cambridge, 1990.
- [8] M. Goresky and R. MacPherson. *Stratified Morse theory*, vol. 14 of Ergebnisse der Mathematik und ihrer Grenzgebiete (3) [Results in Mathematics and Related Areas (3)], Springer-Verlag, Berlin, 1988.
- [9] J.D. Hauenstein. Numerically computing real points on algebraic sets. *Acta Appl. Math.*, 125(1), 105-119, 2013.
- [10] S. Lefschetz. *L'analysis situs et la géométrie algébrique*. (French) Gauthier-Villars, Paris, 1950.
- [11] C. Wampler, B. Larson, and A. Edrman. A new mobility formula for spatial mechanisms. In *Proc. DETC/Mechanisms & Robotics Conf.*, Sept. 4-7, Las Vegas, NV (CDROM), 2007.

LIST OF ACRONYMS, ABBREVIATIONS, AND SYMBOLS

ACRONYM	DESCRIPTION
1-D	1 dimensional
2-D	2 dimensional
3-D	3 dimensional
BR	Bertini Real software package